MéDISIS, l'intégration des analyses de SdF aux processus d'Ingénierie Système Basée sur les Modèles

MéDISIS, integrating dependability analysis to Model-Based System Engineering Processes

Pierre DAVID
Heudiasyc UMR 6599 / UTC
Centre de recherches Royallieu
BP 20529
60205 Compiègne Cedex
pierre.david@hds.utc.fr

Vincent IDASIAK & Frédérec KRATZ
Institut PRISME (équipe MCDS) / ENSIB
88 Boulevard Lahitolle
18020 Bourges cedex
vincent.idasiak@ensi-bourges.fr
frédéric.kratz@ensi-bourges.fr

Résumé

La complexité des nouveaux systèmes ne cesse de s'accroître, en termes d'intégration de multiples technologies, de nombre de composants, ainsi qu'en termes de performances attendues et en contraintes sécuritaires. De plus, les phases de conception de ces systèmes doivent également garantir la tenue de délais stricts pour un coût maîtrisé. Nous avons développé une méthode d'analyse de la sûreté de fonctionnement des systèmes complexes, intégrée aux méthodes d'ingénierie système dirigées par les modèles nommée MéDISIS. Nous montrons l'apport du langage SysML (OMG, 2008) pour l'établissement d'un modèle commun aux différents intervenants du projet, la traçabilité des exigences, et l'automatisation des liens vers les études de sûreté de fonctionnement (David, 2009), (David et al. 2009a), (David et al. 2009b), (David et al. 2010).

Nous présentons dans un premier temps, comment extraire des diagrammes SysML fonctionnels, les informations nécessaires aux études de risques et constituer une AMDEC préliminaire servant de base aux études de sûreté de fonctionnement. Pour cela, des règles d'analyse des modèles SysML sont formulées à partir de l'étude de métamodèles et corroborées par le biais d'études de cas industriels. A partir de ces règles, nous proposons une procédure de traitement des modèles SysML pour la génération d'AMDEC préliminaire. Cette procédure fait appel au retour d'expérience géré à travers une base de données des comportements dysfonctionnels.

Dans un second temps, nous expliquons la mise en place d'une traduction des modèles SysML vers une description utilisant AltaRica Data Flow, destinée à l'analyse quantitative de la SDF (Rauzy et al. 2008). Organisée en deux étapes, la création des modèles en AltaRica Data Flow, comporte la traduction directe de la partie fonctionnelle depuis des modèles SysML. Puis la partie concernant le comportement dysfonctionnel est ajoutée en exploitant les données recueillies lors de la phase d'AMDEC et gérées à travers notre base de données des comportements dysfonctionnels.

Summary

The design processes of new innovative systems must be able to master the increasing complexity brought by the need of integrating multiple technologies, utilized by even more components with high awaited performances. Moreover, stakeholders impose strict delays and costs on their development phase. Many of those systems are intended to possess strong reliability performances. This kind of system must include in their design phase complicated reliability analysis, within a limited time and with limited resources. Therefore, we tackle in this work the construction of a methodology, named MéDISIS, for integrating reliability analysis in the design processes using Model-Based System Engineering techniques.

The main goal of this work is to make the reliability analysis more efficient during the engineering process, by creating methods and tools using or being included in the current design software utilised by system engineers. As we assume that SysML is the best language for Model-Based System Engineering, we developed 3 axes: Using SysML as the language used for the central model of all developments. Extracting the needed information on the dysfunctional behaviour, from functional SysML description studied with classic FMEA-like techniques. Setting translations between SysML descriptions and formal models for reliability studies. In this article, we propose a translation from SysML models to AltaRica Data Flow completed by the exploitation of a knowledge base built from FMEA results.

1. Introduction

Si l'on étudie les systèmes à fortes contraintes de Sûreté de Fonctionnement, on constate qu'ils ne cessent de se complexifier. Cette tendance se vérifie aussi bien pour la réalisation de systèmes de sécurité, que pour celle de transport de personnes ou de systèmes d'exploration spatiale. Cela transparaît notamment à travers la complexification des interfaces et composants remplissant les missions confiées aux systèmes. On note premièrement une augmentation significative de la taille de ces systèmes en terme de nombre de composants utilisés et une hétérogénéité accrue des technologies employées. En effet, la miniaturisation permet l'intégration d'un nombre important d'éléments agrégeant par exemple électronique, mécanique, hydraulique et logiciel.

La démarche d'Ingénierie Système adoptée pour leur conception doit permettre de gérer toutes les tâches nécessaires au développement du système, de la détermination de l'architecture fonctionnelle à réaliser, à la validation des choix opérés et des performances. Les analyses de Sûreté de Fonctionnement prennent une part importante de ce cycle de développement, consomment un temps non négligeable et requièrent une forte expertise du domaine tant les techniques et formalismes à employer sont complexes. Le défi qui se dégage, est donc de favoriser l'interconnexion des activités de conception pures et des étapes d'étude de la Sûreté de Fonctionnement. Pour cela, il faut comprendre les techniques employées pour ces deux facettes de la création des systèmes et en particulier analyser comment unifier et faire coopérer les outils employés par chaque communauté.

En effet, de tels développements ne se font sans l'utilisation de méthodes rigoureuses supportées par les formalismes et outils adéquats. L'utilisation de l'Ingénierie Système basée sur les Modèles est primordiale pour des projets de cette envergure. Nous avons basés notre réflexion sur les méthodes d'Ingénierie Système basées sur les modèles, pour le développement des systèmes complexes à fortes contraintes de Sûreté de Fonctionnement. Nous avons donc cherché à dresser une méthodologie d'intégration des analyses de Sûreté de Fonctionnement au processus d'Ingénierie Système basée sur les modèles, que nous avons intitulée MéDISIS (David, 2009). Le but recherché est celui appelé de leurs vœux par les auteurs de (Rauzy et al. 2008) : « [dissoudre] les frontières entre les environnements de développement virtuel au service des Concepteurs, et les référentiels de simulation accompagnant le travail des Fiabilistes, Logisticiens et Risk Managers, […] pour laisser place à une discipline commune que l'on pourrait qualifier d'Assurance Performentielle… ».

La suite de l'article sera organisée comme suit : nous présentons dans un premier temps les caractéristiques de l'Ingénierie Système Basée sur les modèles (ISBM) et la problématique de la prise en compte des exigences de Sûreté de fonctionnement (SdF) au cœur de tels processus. Nous décrivons alors les grandes lignes de la méthode MéDISIS permettant d'orchestrer la réalisation des étapes d'analyse de SdF au cœur des phases de développement des systèmes. Nous présentons ensuite les traitements mis au point pour optimiser la réalisation d'AMDEC et l'obtention de modèles AltaRica Data Flow dédiés à l'analyse quantitative de SdF.

2. L'ISBM et la gestion des exigences de SdF

2.1 Présentation générale de l'ISBM et ses défis

L'Ingénierie Système Basée sur les documents a longtemps été l'approche usuelle pour l'ingénierie des systèmes physiques. Puis certains domaines l'ont abandonnée au profit d'approche d'ISBM. Les ingénieurs en mécanique et systèmes électriques ont notamment abandonné le dessin technique pour l'utilisation d'outils de modélisation assistée par ordinateur au cours des années 80. Puis le monde du logiciel a adopté et largement amélioré l'ISBM pendant les années 90. L'ISBM se répand maintenant au sein des communautés d'ingénieurs système et devrait devenir le standard de développement des prochaines années, comme le souligne les prévisions de l'INCOSE (INCOSE, 2007), (Friedenthal et al. 2007).

L'ISBM est l'utilisation formalisée de modèles comme supports aux activités de spécification, conception, analyse, vérification et validation des systèmes, débutant de la phase de conception préliminaire et se poursuivant à travers le développement jusqu'aux dernières phases de vie du système (INCOSE, 2007). Les techniques d'ISBM mettent l'accent sur l'évolution et la progression dans le niveau de détail lors de l'étude du système. L'ISBM est censée faire progresser l'Ingénierie Système dans de nombreux domaines comme la communication entre les différents acteurs d'un projet, la précision de l'analyse, l'intégration des résultats ou la réutilisation des connaissances produites, tous ces aspects participent à une réduction substantielle des risques liés au développement.

Pour être mise en place, l'ISBM doit être déployée par une méthodologie. (Estefan, 2008) indique qu'une méthodologie d'ISBM peut être décrite comme une collection de processus, méthodes et supports utilisés pour assister la réalisation de l'IS dans un contexte « basé modèle » ou « dirigé par les modèles ». Cette définition fait apparaître la notion centrale de modèle, en prise directe avec la ou les méthodes membres de la méthodologie. L'ISBM possède deux types d'entités fondamentales : le modèle du système et les bases de connaissances contenant le modèle et ses dérivés. La gestion du modèle et des connaissances qu'il génère impose l'utilisation d'un Environnement de Développement Système (EDS) réunissant la suite d'outil employée au cours de l'ISBM.

(Friedenthal et al. 2008) propose la définition suivante d'un modèle : « Un modèle est une représentation d'un ou plusieurs concepts pouvant être réalisés dans le monde physique. Il décrit généralement un domaine d'intérêt. » Les modèles utilisent de nombreuses formes : graphiques, mathématiques, représentations logiques, prototypes physiques. Une caractéristique clé des modèles est qu'ils sont une abstraction ne contenant pas tous les détails de l'entité, cela introduit la notion extrêmement importante de modélisation dans un domaine d'intérêt, une vue. Une vue est, donc, une présentation partielle des connaissances sur l'élément modélisé. Les vues permettent de partitionner l'étude d'un système afin de proposer à l'analyste le sous-ensemble d'information nécessaire et suffisant à sa prise de décision. L'usage des vues permet donc de découper le problème global d'analyse ou de conception du système, en sous-problèmes plus aisément solvables ; les processus de l'ISM se chargent d'assurer la cohérence des activités.

En ISBM, le modèle est constitué d'éléments souvent graphiques, qui représentent les exigences, l'architecture, le comportement, les cas de tests, les justifications et toutes les relations entre ces éléments. Le modèle est d'abord employé pour concevoir le système conformément à ses spécifications. Il met en avant la décomposition du système et la distribution d'exigences qui en découle. Le modèle sert alors de support de transition de l'échelle système au niveau composant, préparant ainsi les développements particuliers induits. L'exploitation du modèle peut par exemple se traduire par la génération de spécifications de réalisation des sous composants. Il sert également de base à la remontée des performances des sous-blocs et l'évaluation de leur participation aux performances globales. Enfin, ce modèle sert de base aux analyses et simulations du système, ainsi qu'à la génération de documents contractuels ou normatifs. Si le modèle est exécutable, ces fonctions peuvent être conduites par l'outil de modélisation, sinon des passerelles doivent exister entre les différents outils constituant l'EDS. Ces étapes induisent régulièrement des générations assistées de modèles. La place centrale occupée par le modèle au sein du développement des systèmes est illustrée par la figure 1. Le modèle y est le point de rencontre et la passerelle entre exigences, analyses, retour d'études, conception détaillée et production documentaire. Rappelons qu'il sert aussi bien à préparer les études particulières qu'à répercuter leurs résultats et conclusions. Nous avons choisi le langage SysML comme support d'expression du modèle. Ce langage nous apparaît comme la meilleure alternative actuelle pour la constitution du langage central à une méthodologie d'ISBM. Nous justifierons ce choix et présenterons le langage dans le paragraphe suivant.

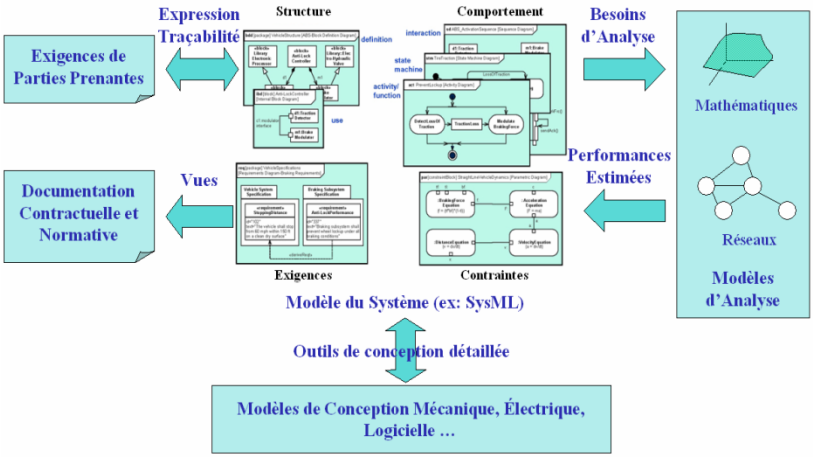

Figure 1. Le modèle du système au centre de l'IS adapté de (Friedenthal et al. 2008)

Les méthodologies d'ISBM doivent une grande part de leur efficacité à la puissance de l'EDS outillant la démarche. L'EDS est constitué de l'ensemble des outils utilisés pour réaliser les traitements mentionnés sur la figure 1. (Friedenthal et al. 2008) donne une définition complète des EDS : un EDS fait référence aux outils et bases de connaissances utilisés pour le développement d'un système. Les outils communément utilisés dans un EDS permettent : la modélisation, l'analyse des composants physiques et logiciels, le test et le management de projet. Les bases de connaissances doivent alors permettre le transfert et la cohérence des informations. Nous proposons l'ajout cohérent d'outils spécifiques d'analyse de SdF ; MéDISIS rassemble les nouveaux processus d'ISBM associés à leur emploi.

2.2 Un langage support à l'ISBM : SysML

Le langage SysML est actuellement le langage le mieux adapté au paradigme de l'ISBM. Il est largement diffusé comme langage de base pour des méthodes d'ISBM développées par nombre d'industriels ou associations d'ingénieurs. L'usage de ce langage se retrouve au sein de méthodes comme Harmony d'IBM Telelogic ((Douglass, 2005), (Hoffmann, 2006)), OOSEM (Object-Oriented, Systems Engineering Method, (Lykins et al., 2000), (Friedenthal et al. 2008)) ou RUP SE (Rational Unified Process for Systems Engineering (Cantor, 2003)). Le langage

SysML utilise l'approche Orientée-Objet et constitue un langage de modélisation graphique sans a priori technologique sur le système modélisé. Il est conçu à la fois pour aider à l'analyse, la spécification, la conception, la vérification et la validation des systèmes complexes (OMG, 2008). Le langage offre la possibilité de modéliser des composants physiques ou logiciels, des acteurs humains, des flux de données, des procédures et tout ce qui peut composer un système ou un système de systèmes. Il a été prévu pour fournir des descriptions préparant l'emploi de formalismes dédiés à un domaine d'étude ou un type d'analyse spécifique. Ce pouvoir d'expression est important dans l'utilisation de SysML, comme nous l'avons illustré par la figure 1. SysML permet de modéliser les systèmes, leurs composants et la majorité des entités qui leur sont relatifs, il fournit des artefacts modélisant :

- La structure, les interconnexions et classifications des composants.
- Les comportements basés sur des fonctions, des états et des messages.
- Les contraintes sur les propriétés physiques et les performances.
- Les exigences, leurs interdépendances, leur affectation à la structure et leur méthode de vérification et validation.
- Les allocations entre comportement, structure, exigences et contraintes.

L'objectif des créateurs de SysML était de proposer aux ingénieurs système un langage de modélisation simple et à la fois suffisamment expressif. C'est pourquoi la spécification est réduite à un petit nombre de diagrammes recouvrant les différentes dimensions de la description des systèmes. L'organisation des diagrammes utilisables est présentée à la figure 2. Ils sont regroupés par axe de modélisation, on distingue ainsi les diagrammes relatifs à l'expression de la structure et ceux relatifs à la description du comportement. On peut noter que la gestion des exigences a nécessité l'introduction d'un troisième axe, reflétant les besoins de traçabilité des demandes issues du cahier des charges durant le processus d'IS. Les principales caractéristiques d'un modèle utilisant SysML sont énoncées dans la spécification (OMG, 2008). Ce modèle est :

- *Soumis à des exigences* : SysML propose des artefacts permettant de spécifier, d'organiser, de gérer des exigences.
- *Composé* : les systèmes sont modélisés en termes d'associations de composants.
- *Hiérarchisé* : le modèle peut être organisé en répertoires dénotant le niveau de granularité des descriptions. Les artefacts peuvent posséder des relations de filiation.
- *Interopérable* : les modèles SysML sont destinés à être échangeables et partageables entre différents outils de l'EDS.

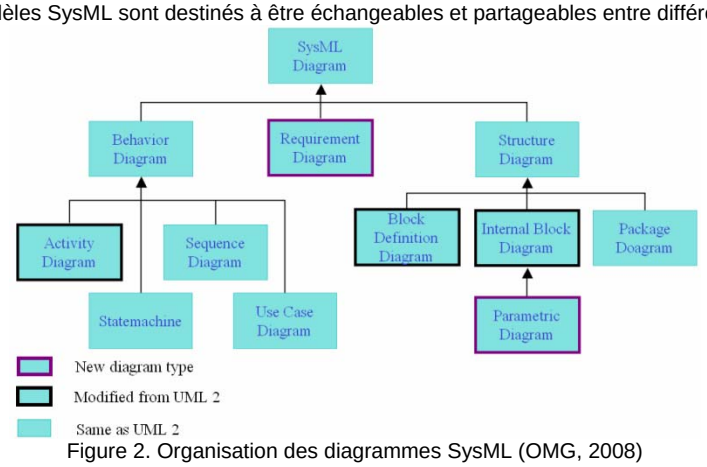

Figure 2. Organisation des diagrammes SysML (OMG, 2008)

2.3 Intégrer les analyses SdF à l'ISBM et à l'EDS

Au cours du processus d'ISBM, le modèle commun d'ingénierie utilisé doit permettre de connecter et initier les différentes activités d'analyse et de conception détaillée nécessaires à la production du système désiré. Pour l'analyse de systèmes à fortes contraintes de SdF, le processus passe par d'importantes analyses du comportement du système, incluant l'observation des activités fonctionnelles et dysfonctionnelles du système. La validation formelle et la recherche de scénarios de défaillance sont les deux grands axes de travail relevés dans la littérature.

Le processus d'agrégation des connaissances nécessaire à une analyse de SdF peut être décomposé en deux phases .

Dans un premier temps (phase qualitative), il nécessite la création et l'incorporation de nouvelles connaissances dépassant la vue fonctionnelle du système. Il s'agit du recensement du comportement dysfonctionnel de chaque élément du système. Cette partie requiert l'emploi de méthodes de création ou de recouvrement d'informations, procédant par analyse de la description fonctionnelle du système et utilisation de jugement d'experts. Ce sont d'une part l'analyse fonctionnelle menée dans les premières phases du processus d'IS, et la réalisation d'Analyse des Modes de Défaillance de leurs Effets et Criticité (AMDEC) et d'Analyse Préliminaire des Risques (APR) d'autre part.

Dans un second temps (phase quantitative), l'analyse de SdF du système se poursuit par l'exploitation des connaissances précédemment rassemblées. Le but est alors la qualification et la quantification des aspects plus généraux du modèle, tel que l'atteignabilité d'un état redouté ou le calcul de la disponibilité globale du système. Durant cette phase, la connaissance nécessaire a déjà été produite, il reste à l'analyser pour produire les vérifications souhaitées. Ceci nécessite la création d'un modèle formel du système reflétant les facettes fonctionnelles et dysfonctionnelles de celui-ci. Les analyses produites à partir de ces modèles procèdent de deux manières distinctes : l'analyse par scénarios de fonctionnement du système (scénarios unitaires ou ensembles de scénarios), la preuve mathématique de propriétés du modèle. De nombreux outils, méthodes, langages et formalismes ont été développés pour réaliser cette partie de l'analyse. On peut distinguer deux approches principales, de par leur diffusion : l'utilisation, les automates à états finis et les Réseaux de Petri (RdP).

Nous avons bâti une méthode pour intégrer efficacement le sous-processus d'analyse de SdF des systèmes au processus plus global d'ISBM. Cette méthode intitulée MéDISIS (Méthode D'Intégration des analyses de Sdf à l'Ingénierie Système) est le résultat de la thèse (David, 2009) et a fait l'objet de plusieurs publications (David et al. 2009a), (David et al. 2009b), (David et al. 2010). Son but est de proposer un enchaînement de traitements de l'information et des connaissances, incluant la création, l'expression, l'analyse, la pérennisation et la réutilisation répondant aux besoin décrits précédemment. Ces traitements doivent être supportés par des outils et répertoires formant un EDS cohérent. Nous avons souhaité définir cette méthode pour l'étude de systèmes complexes et multi-technologiques. Nous faisons également le choix de réutiliser les outils des industriels tout en proposant une méthode ne ciblant, en particulier, aucun d'entre eux. Ces travaux doivent être utilisables sur un large ensemble d'outils de conception. Les objectifs à atteindre par la méthode sont les suivants :

- Faciliter la transmission de connaissances entre équipes.
- Accélérer la réalisation des études de SdF.
- Organiser l'exploitation commune des connaissances sous formes de modèles.

- Permettre la réutilisation des connaissances entre projets.
- Identifier les besoins d'analyse et réaliser le suivi de leurs résultats.
- Améliorer la cohérence et la qualité des analyses SdF.

Nous considérons que la source d'information initiale à MéDISIS est un modèle SysML purement fonctionnel, résultat de l'analyse fonctionnelle du système étudié. La méthode a pour objectif de maximiser l'efficacité des analyses SdF durant la conception, en les rendant plus rapides, cohérentes et pertinentes. La méthode est soumise à différentes contraintes dues à son positionnement et à son rôle dans le processus d'ISBM. Elle doit être utilisable à divers niveaux de détails du système, être itérative (permettre la descente dans ces niveaux de détails) et répétable. Les connaissances collectées au cours de l'étude concernant les composants, sous-systèmes et systèmes doivent être réutilisables afin de remplir le rôle des approches modernes d'IS. Pour cela, une Base de données des Comportements Dysfonctionnels (BCD) constitue l'élément central de MéDISIS. Cette BCD regroupe les connaissances collectées sur les mécanismes de défaillances des entités composant le système analysé. MéDISIS inclut des services de génération automatique d'analyse ou de modèle, utilisés pour réaliser chaque étape de la démarche. La figure 3 schématise MéDISIS et présente l'enchaînement des procédures effectuées et l'utilisation de la BCD.

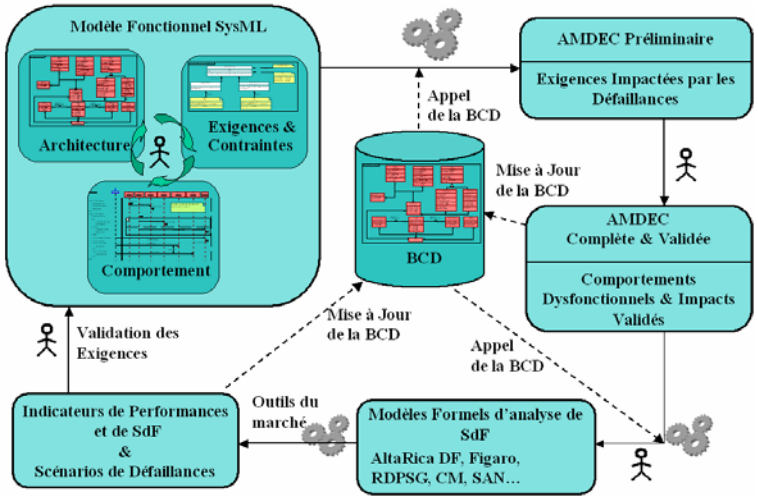

Figure 3. La méthode MéDISIS

MéDISIS est une méthode déductive et incrémentale se déroulant en 3 phases majeures :

- Recherche du comportement dysfonctionnel des composants et des influences entre composants voisins. Identification des exigences impactées. Cette phase est finalisée par une AMDEC munie d'une assistance automatisée et correspond à la phase qualitative.
- Construction assistée d'un modèle utilisant une représentation formelle du comportement fonctionnel et dysfonctionnel du système (phase quantitative).
- Analyse outillée des modèles formels, validation des exigences, retour sur la conception.

L'analyse est conduite sur le système ou ses sous-systèmes. Le lancement est précédé d'une définition des limites de l'étude, facilitée par l'organisation naturelle des modèles SysML. On considère qu'un modèle présentant l'architecture et le comportement purement fonctionnel est disponible, ainsi qu'une première répartition des exigences. Ce modèle est issu des premières phases du processus d'ISBM. Un premier recensement des exigences de SdF peut avoir été réalisé au préalable en utilisant par exemple une Analyse Préliminaire des Risques.

La première phase, détaillée dans [12, 14], consiste à la réalisation d'une AMDEC du système. L'AMDEC permet un recensement rapide des MdD de chaque composant. Elle permet de réaliser une détection systématique des risques pesant sur le système très tôt dans la conception. Ces risques sont analysés et classifiés afin de diriger la suite de l'étude. Cette première phase est critique pour la suite de l'analyse car elle est chargée de définir le comportement défaillant des composants qui sera quantifié et qualifié pour le système global dans la suite de l'étude. L'AMDEC est réalisée en conjonction avec l'utilisation de la BCD. Cette base permet de faire appel au Retour d'EXpérience (REX) sur des systèmes ou composants similaires déjà utilisés. Ce REX est primordial pour les AMDEC où seuls les jugements d'experts s'expriment. Dans le cadre de MéDISIS, cette connexion à la BCD est double. En effet, les résultats de l'AMDEC validée par les experts sont également réintroduits dans la BCD pour préciser et enrichir ses connaissances. Cette phase est découpée en deux étapes : la création d'un rapport AMDEC préliminaire destiné à soutenir les experts et éliminer la lourdeur de l'étude, et une finalisation manuelle de l'AMDEC par les experts dont le jugement et la prise de décision finale ne peuvent être automatisés.

La seconde phase de l'étude consiste à construire les modèles qui vont permettre une analyse formelle de la SdF du système. Ces études nécessitent l'emploi de langages et formalismes complexes et souvent difficiles à mettre en place. Il est donc judicieux, comme le mentionne également (Dumas et al. 2008) de fournir une aide à la génération de ces modèles au cours du processus d'ISBM. Ces modèles doivent reprendre la définition de la structure et des comportements fonctionnels et dysfonctionnels des composants. Ils redéfinissent donc une partie déjà couverte par le modèle du concepteur. Il est alors intéressant d'assister la création de ce modèle en traduisant le modèle SysML fonctionnel. Cette étape permet d'accélérer l'analyse de SdF et d'assurer une plus grande cohérence entre les modèles du projet manipulés dans l'EDS. Cette étape est complétée par l'utilisation de la BCD. Elle permet de faire appel à des éléments de modèles dysfonctionnels déjà connus, qui viennent se greffer à la partie fonctionnelle. La BCD peut alors être vue comme une bibliothèque de modèles dysfonctionnels. Dans notre travail, nous nous sommes focalisés sur l'emploi du langage AltaRica Data Flow (DF) dont la sémantique proche de l'OO permet une traduction efficace de modèles SysML, comme nous l'évoquons dans (David et al. 2009b). Ce langage particulièrement bien outillé permet de créer des modèles exploitables pour le calcul d'indicateurs quantifiés et la recherche de scénarios de défaillance.

La dernière phase de MéDISIS est l'exploitation du modèle ainsi créé pour analyser le respect des exigences formulées sur le système. Les conclusions de cette exploitation permettent de juger l'architecture proposée et de valider les choix de conception. Au cours de cette étude sont obtenus des résultats d'analyse permettant de fournir les documents contractuels ou nécessaires à la certification, attendus lors du développement. Les conclusions orientent également l'effort d'analyse en pointant par exemple la nécessité d'étudier plus en détail un sous-système crucial pour la SdF du système. La cohérence des modèles employés permet d'obtenir un suivi rigoureux des exigences et une bonne focalisation des tests. L'usage de la BCD centralise la gestion des connaissances créées et réutilisées tout au long de MéDISIS.

3. L'assistance à la création d'AMDEC par l'analyse automatisée de modèles SysML

L'automatisation et la réalisation d'AMDEC sur des modèles utilisant un langage spécifié sont des leviers importants pour la réussite de l'intégration des AMDEC à l'ISBM. L'AMDEC pourra bénéficier des mêmes avantages que ceux relevés lors de l'utilisation de méthodes basées sur les modèles, à savoir : meilleure qualité d'étude, meilleure communication entre équipes, meilleure productivité et meilleur transfert des connaissances. En effet, par l'analyse de modèles purement fonctionnels exprimés en SysML, il devient possible d'automatiser des étapes comme la recherche de composants et de guider l'analyse de l'expert lors de la rédaction de l'AMDEC finale. L'objectif de l'assistance à la création d'AMDEC peut être résumé comme la volonté de consacrer 80% du temps de l'expert aux 20% de l'étude les plus délicats.

Pour réaliser cette assistance il nous faut comprendre au préalable le raisonnement employé pour la réalisation d'une AMDEC. L'analyse de cette méthode nous a conduits à proposer le métamodèle de la figure 4 illustrant l'organisation des connaissances rassemblées lors d'une AMDEC.

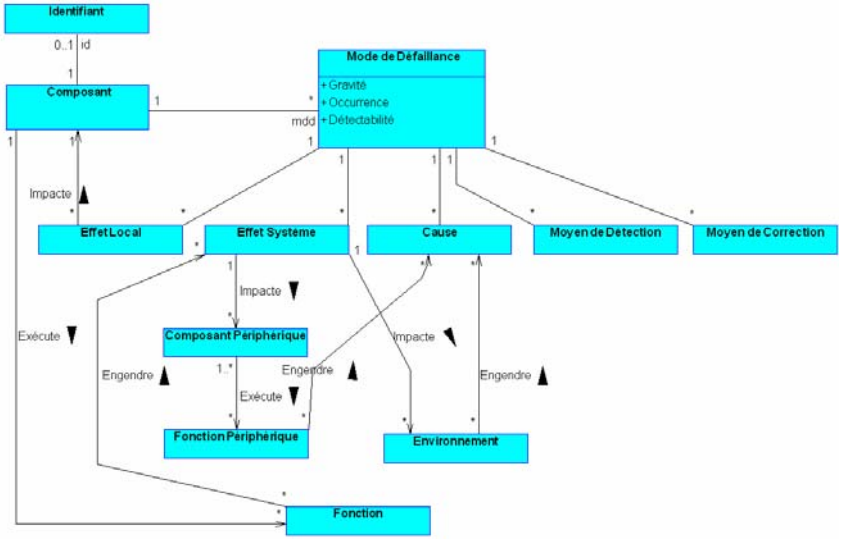

Figure 4. Métamodèle des données recherchées par l'AMDEC

Notre travail a consisté à montrer comment les éléments nécessaires à l'obtention de cette structure de données pouvaient être identifiés dans les diagrammes SysML des concepteurs.

Lors d'une AMDEC, le processus de prise de décision est mené pour établir des relations de cause à effet à partir de vues structurelles et fonctionnelles du système pour générer un tableau AMDEC. L'étude AMDEC passe par les étapes suivantes :

1. Recensement des composants à étudier.
2. Pour chaque composant, rechercher les MdD.
3. Pour chaque MdD, rechercher les causes possibles.
4. Pour chaque couple MdD/causes possibles, rechercher les effets possibles sur les composants périphériques.
5. Pour chaque effet, réaliser la cotation du risque (RPN), proposer des moyens de détection et spécifier les actions à mener pour diminuer le risque.

Toutes ces étapes engendrent la constitution d'un tableau regroupant les résultats de l'AMDEC. L'étape 1 se base sur une vue structurelle du système, les éléments à identifier sont les composants du système. Cette vue est couverte par l'axe structurel de SysML. L'étape 2 fait appel aux connaissances des experts (organisées ou non dans des bases de données), les éléments relevés sont le nom des MdD, cette étape est réalisée dans MéDISIS par la connexion à la BCD. L'étape 3 est la première faisant appel au raisonnement de causalité, elle est basée sur l'expérience des experts et l'analyse des vues du système. En effet, comme nous le constatons dur la figure 4 la défaillance peut provenir de l'entourage du composant affecté. Pour cela, il convient d'identifier les interfaces du composant ouvertes à l'influence des composants périphériques. Pour l'étape 3, on recense les causes connues par les experts, les interfaces offertes par le composant et les composants périphériques, puis on conserve ceux dont la fonction peut provoquer le MdD. Pour cela les diagrammes de l'axe structurel de SysML sont analysés pour découvrir les chemins de propagation. L'étape 4 fait également appel au raisonnement causal : le MdD du composant induit un changement de l'exécution de ses fonctions. Ces fonctions dégradées produisent leurs sorties, transmises par les interfaces du composant aux composants périphériques. La réception de ces flux potentiellement erronés provoque des changements d'état des composants périphériques et de leurs traitements, ces informations sont les effets du MdD. Les éléments à produire ici sont les comportements des fonctions altérées, les flux altérés, les interfaces les relayant et les composants périphériques touchés ainsi que leur comportement modifié. Pour cela on étudie plus en détail les diagrammes de l'axe comportemental de SysML, ces derniers montrent les enchaînements de traitement réalisés ainsi que les paramètres impactés. Enfin, l'étape 5 fait appel aux jugements et connaissances des experts pour produire le RPN et communiquer les solutions adéquates pour la gestion du MdD.

Nous avons ainsi énoncé 7 règles d'analyse des modèles SysML utilisées pour la préparation des AMDEC[1] :

Règle N°1 : Les composants identifiés par l'AMDEC sont les *parts* du modèle. Leur nom donne le nom du composant. Leur type est donné par le *block* dont le *part* est une instance.

Règle N°2 : Les fonctions de chaque composant sont identifiées en relevant :

- Les *operations* du *block* typant le *part*,
- Les *actions* allouées par une *swimlane* au *part* ou au *block* le typant,
- Les *actions* de type « *call operation action* » appelant une *operation* du *block* typant le *part*.

Règle N°3 : Les interfaces et relations structurelles sont identifiées par l'analyse des *ports*. Les *ports* reliés entre eux par les connecteurs et les *parts* qui les possèdent sont respectivement les interfaces et les composants périphériques. Le sens de l'interface est déduit en considérant la propriété *direction* des *flow ports*.

[1] Les concepts du langage SysML sont indiqués en *italique*.

Règle N°4 : Les composants périphériques sont, en complément de ceux identifiés par la règle N°3, les *parts* recevant/émettant des *messages* du/vers le *part* étudié et les *parts* responsables d'*actions* en relation avec celle du *part* étudié.

Règle N°5 : L'analyse de la propagation des défaillances des composants à travers sa représentation fonctionnelle, s'appuie sur les diagrammes de l'axe comportemental, afin de rechercher les causes et effets de ces défaillances. La propagation est réalisée dans le sens permis par les interfaces identifiées par la règle N°3 et les interactions relevées par la règle N°4.

Règle N°6 : Les exigences relatives aux composants sont identifiées en considérant les relations *satisfy* entre les *parts* et les *requirements*, entre les *typing blocks* ou *owning blocks* et les *requirements*, et entre les *properties* des *parts* ou *typing blocks* et les *requirements*. Les *requirements* dérivés (*derived*) ou possédant un lien de *containment* avec les *requirements* satisfaits par le *part* sont également liés au composant.

Règle N°7 : Les contraintes en relation avec les composants sont identifiées en considérant les *constraint properties* dont les *constraint parameters* sont liés aux *values properties* des *parts*. Ce sont en particulier toutes les *constraint properties* du *typing block* des *parts*.

Ces règles fournissent un support à l'analyse du système à travers son modèle SysML en vue de la réalisation de son AMDEC. Ces dernières permettent d'isoler dans le modèle complet les éléments d'informations nécessaires pour la réalisation des étapes successives constituant le raisonnement produit lors d'une AMDEC. Le tableau 1 résume les phases d'utilisation de chaque règle vis à vis des étapes de l'AMDEC.

Tableau 1. Analyse de modèle SysML et création d'AMDEC

Etape de réalisation AMDEC / Règle d'analyse SysML/elts MéDISIS	**1 : Recensement composants**	**2 : Recensement MdD**	**3 : Recherche Effet/Cause**	**5 : Cotations Actions correctives**
Règle 1	●			
Règle 2		●	●	
Règle 3			●	
Règle 4			●	
Règle 5			●	
Règle 6			●	●
Règle 7		●	●	
BCD		●		●
Jugement d'expert		●	●	●

Ces règles d'analyse du modèle peuvent être implémentées dans l'outil de modélisation afin d'automatiser cette analyse. Une telle solution permet de créer automatiquement un rapport AMDEC préliminaire servant de guide à la création de l'AMDEC finale. Ce rapport préliminaire concerne la réalisation des tâches les plus rébarbatives de l'étude et permettent à la réflexion de se porter sur les activités les plus complexes du raisonnement. Le choix de la méthode d'implémentation dépend de l'EDS choisi. Cependant, nous avons exploré au cours de nos travaux quelques alternatives qui nous permettent de commenter le choix des techniques à employer. Trois possibilités majeures sont envisageables : l'utilisation des fichiers d'échanges au format XMI, l'utilisation des fichiers natifs des outils de modélisation ou l'utilisation des services de création de code propres à ces mêmes outils. Les deux premières solutions peuvent s'appuyer sur l'utilisation de langages de transformation de modèles comme ATL (ATLAS Transformation Language) (Jouault et al. 2005), (Chenouard et al. 2008) ou l'outil KerMeta (Muller et al. 2005). L'utilisation des fichiers XMI montre des limites en termes d'informations accessibles. En effet, l'intégralité du modèle n'est pas accessible dans ce format de sortie, néanmoins ce type d'implémentation est la plus pérenne car elle exploite un type de fichier commun à tous les outils d'ingénierie exploitant SysML. La seconde possibilité possède le plus fort potentiel de traduction mais exige une connaissance complète des métamodèles des fichiers traduits, or ces derniers sont difficilement accessibles et exprimables pour la création de ce type d'outil. Enfin la création d'un service au cœur de l'outil ne conduira qu'à l'obtention de solution particulière à une version d'un logiciel de modélisation SysML, obligeant les mises à jour des services de traduction au cours de l'évolution des solutions de modélisation.

4. De SysML à AltaRica Data Flow

La constitution d'un modèle formel du système reflétant son comportement dysfonctionnel est un point crucial pour ses phases de validation. Les qualités que l'on cherche à atteindre au cours de cette étape sont en tous points identiques à celles recherchées pour le processus AMDEC : traçabilité envers le modèle central du système et ses exigences. La volonté principale est donc d'assister, voire d'automatiser la création du modèle formel à partir du modèle fonctionnel du système et des connaissances du comportement défaillant. Nous proposons pour cela de réaliser une traduction des modèles SysML vers le langage AltaRica Data Flow, dans une approche similaire à (Dumas et al. 2008) qui ont exploré la conversion de modèles AADL[2] en descriptions AltaRica Data Flow.

Nous nous plaçons dans une phase de l'étude du système où l'AMDEC a été réalisée. Par cette analyse, constituant le lancement de MéDISIS, les analystes ont obtenu les Modes de Défaillance de chaque composant et ont pu cibler leurs effets potentiels sur le système. L'étude à mener est alors de qualifier et quantifier les effets de ces défaillances à l'échelle du système global. Ces validations viennent en réponse aux exigences formulées par les parties prenantes ou en prévision d'une demande de certification. Cette étape est la seconde phase de MéDISIS, nous avons ici étudié comment intégrer ce type d'analyse, faisant appel à des langages et représentations formels, à l'Ingénierie Système Basée sur les Modèles et son Environnement de Développement Système. Il s'agit donc d'utiliser toutes les connaissances contenues dans le modèle fonctionnel commun au processus d'Ingénierie Système, mais également d'exploiter les connaissances relevées sur le comportement dysfonctionnel au cours de l'AMDEC.

Dans cette partie, nous étudions l'aide à la saisie du modèle dysfonctionnel en AltaRica Data Flow, en ciblant la traduction des éléments du modèle SysML en AltaRica Data Flow. Cette construction se fait en trois phases et sous deux vues : la vue fonctionnelle et la vue dysfonctionnelle. Nous détaillons la retranscription de la structure du système, puis de son comportement fonctionnel et enfin l'adjonction du comportement dysfonctionnel. Pour les deux premières parties, nous travaillons à partir du modèle SysML issu de l'analyse fonctionnelle. La

[2] AADL : Architecture Analysis & Design Language [SAE 2009].

partie dysfonctionnelle est constituée en exploitant la BCD constituée à partir de l'expérience de l'institution développant le système et surtout par les éléments recensés lors de l'AMDEC.

Afin de résoudre les problèmes de complexité de modélisation des grands systèmes multi-technologiques, SysML propose de suivre la voie d'une représentation hiérarchique et compositionnelle. Le concept de *block* se retrouve donc au centre de la représentation. Dans un second temps, la spécification met l'accent sur la représentation des liens constituant le maillage de l'architecture étudiée. Les concepts de flux se sont donc vus attribuer une modélisation structurée par leur typage, orientation et valeurs. Enfin, SysML propose de nombreux concepts de modélisation offrant la possibilité de caractériser les changements d'états et le séquencement des traitements réalisés. Ces stratégies pour aborder la modélisation d'un système se retrouvent dans le langage AltaRica DF, dont la notion de composants s'échangeant des flux influencés par et influençant leur comportement est fondatrice. Nous avons ainsi pu proposer une méthode de traduction entre ces modèles pouvant aller jusqu'à l'automatisation du passage d'une représentation à l'autre.

La construction du modèle AltaRica est effectuée en deux étapes et a été discutée dans (David et al. 2009b). Dans un premier temps, le modèle fonctionnel est constitué. Ensuite, la partie dysfonctionnelle est ajoutée sur le modèle précédemment constitué dans lequel apparaissent les composants physiques de l'architecture et leur comportement nominal. Cette seconde étape est réalisée grâce aux informations glanées lors de l'étude AMDEC et par l'utilisation de la BCD commune au processus d'ingénierie global. Cette construction utilise clairement les modèles et données de l'EDS, garantissant ainsi une traçabilité et une exploitation maximale de ces connaissances. Pour chaque élément de la transformation de modèles, nous identifions les éléments du modèle SysML à prendre en compte et leur expression en AltaRica. Pour chacune des phases de la transcription, nous analyserons les restrictions ou adaptations nécessaires en matière de modélisation SysML pour automatiser le processus.

Les tableaux qui suivent expriment les correspondances existant entre les deux langages. Le tableau 2 présente l'obtention des concepts représentant la structure du système, le tableau 3 s'intéresse au comportement fonctionnel. Nous recensons les concepts à modéliser pour représenter le système et nous donnons les artefacts employés dans les deux langages pour refléter ces derniers.

Tableau 2 Traduire la structure du système

Concept	AltaRica DF	SysML
Type de composant / Instanciation	Node Field : sub	Block Part
Variable d'état / Type	Field state Bool, integer, float, domain	Value properties Value Type
Variable de flux / Type / Direction	Field : flow Bool, integer, float, domain In ou Out	Value properties Value Type ∅
Port (Interface) / Type / Direction	Une variable de flux est aussi un port	Flow/standard port Value Type/Block Flow port Direction/Interface
Connexion Inter-composants	Assertion : égalités entre variables de flux	Connectors des Internal Block Diagrams

Cette mise en correspondance des deux langages fait apparaître de nombreux points de recouvrement entre eux. Néanmoins, certains points de la traduction nécessitent l'utilisation de méthodes de traduction pour obtenir le modèle complet en AltaRica DF. De manière générale, SysML dispose d'artefacts plus nombreux et laisse une plus grande latitude au concepteur dans le typage des éléments du modèle. Pour obtenir une traduction directe il est nécessaire de définir soit des routines d'interprétation du modèle SysML, soit de définir des règles d'utilisation de SysML lors de la saisie du modèle. Nous avons pu constater que de telles restrictions loin d'alourdir la modélisation SysML, rendait celle-ci plus cohérente d'un point de vue de l'ingénierie globale. Ces règles de saisie concernent par exemple la déclaration de liens explicites entre variables et interfaces convoyant le phénomène physique qu'elles caractérisent. De même, en ce qui concerne la description du comportement fonctionnel (tableau 3), il est nécessaire de déclarer des allocations entre les composants et les comportements qu'ils exécutent.

Tableau 3. Traduire le comportement fonctionnel

Concept	AltaRica DF	SysML
Comportement interne	Assertion	Operations Constraint properties
Comportement événementiel	Transition : guard	Ordonnancement messages Constraint properties Gardes d'activités
	Transition : event	Opérations Activités Actions
	Transition : affectation	Constraint properties
Synchronisation des comportements	Field : sync	Interaction Activités

L'adjonction de la partie dysfonctionnelle peut être gérée de deux façons, pouvant également être complémentaires. La première solution est de transmettre le « squelette » fonctionnel du modèle, comportant la définition de la structure et du comportement fonctionnel, aux experts de SdF et de confier à ces derniers la saisie de la partie dysfonctionnelle. Cette saisie reprend les techniques employées actuellement à l'exception près que la partie commune au modèle du concepteur est déjà traduite et a priori cohérente, car résultat d'une traduction automatique. La deuxième solution est d'ajouter de façon automatique certaines parties du comportement dysfonctionnel. Pour cela, il faut disposer au préalable d'une base de données de type BCD, comme nous le préconisons dans MéDISIS. Cette base doit être formée pour accueillir les résultats de l'AMDEC et le pérenniser en vue de leur utilisation pour la création du modèle d'analyse de SdF. L'initialisation d'une telle base a été discutée dans (David, 2009) et (David et al. 2009a).

Tableau 4. Traduire le comportement dysfonctionnel

Concept	AltaRica DF	SysML/MéDISIS
Condition de défaillance	Transition : guard	Jugement d'expert
Déclenchement	Transition : event	BCD :
Premier effet	Transition : affectation	Diagrammes paramétriques
Comportement	Assertion	Value properties
Caractéristiques	Field : state Field : extern : law	Operations Allocations Statemachines

La traduction entre modèles SysML et AltaRica DF que nous pouvons fournir permet de retirer différents avantages pour le processus d'analyse de SdF. Dans un premier temps, elle permet d'adosser le modèle d'analyse SdF au modèle commun de l'EDS, pour ainsi disposer d'une représentation cohérente avec le système développé. Deuxièmement, une telle transformation offre l'occasion de créer rapidement une

majeure partie du modèle formel nécessaire aux validations inhérentes à la SdF. Enfin, cette équivalence entre SysML et AltaRica DF autorise le développement d'une base de connaissances commune sur les dimensions dysfonctionnelles du système, capable d'interagir avec les modèles des concepteurs et des experts chargés de la validation. Cette traduction dispose des mêmes possibilités d'implémentation que le service de création d'AMDEC. Nous avons en outre noté que les règles de modélisation SysML à mettre en place pour faciliter la traduction se montraient cohérentes avec celles énoncées pour l'analyse AMDEC et participaient également à obtenir un modèle cohérent et expressif pour l'ingénierie.

5. Conclusion

Nous avons défini la méthode MéDISIS organisant l'analyse de SdF, de la recherche des défaillances particulières des composants, à l'évaluation du comportement dysfonctionnel du système global. Nous proposons à travers elle une succession de traitements des modèles et de l'information recueillie sur le système, organisant l'analyse de SdF du système durant sa conception. MéDISIS peut être qualifiée de méthode déductive et incrémentale capitalisant les résultats d'analyses au sein d'une Base de données des Comportements Dysfonctionnels (BCD) et exploitant les modèles SysML directement issus de la modélisation fonctionnelle. MéDISIS se déroule en trois étapes :

- Recherche, par une AMDEC munie d'une assistance automatisée, du comportement dysfonctionnel des composants et des exigences impactées.
- Construction assistée d'un modèle du comportement fonctionnel et dysfonctionnel du système, en utilisant une représentation formelle.
- Analyse outillée des modèles formels, validation des exigences, retour sur la conception.

Afin d'optimiser l'application de MéDISIS pour des projets de développement utilisant le langage SysML, nous avons analysé les mécanismes de réalisation de ces différentes phases.

Ainsi, nous avons défini des règles d'analyse de modèles SysML pour l'obtention d'AMDEC composant. L'étude réalisée au niveau des métamodèles permet d'obtenir une systématique de traitement applicable à tout type de modèles SysML, quelles que soient les habitudes de modélisation du concepteur. Ces règles sont utilisées pour dégager les données nécessaires à l'établissement d'une AMDEC. La finalisation nécessite tout de même l'intervention d'un expert. Nous facilitons cette dernière en proposant une AMDEC préliminaire, automatiquement générée en appliquant les règles de traitement.

Nous avons développé une solution de traduction de modèle venant soutenir la seconde phase de MéDISIS. Nous nous sommes focalisés sur la création assistée de modèles d'analyse de SdF construits en AltaRica Data Flow. Cette traduction de SysML vers AltaRica Data Flow permet d'intégrer l'usage de ce formalisme au processus d'Ingénierie Système, en apportant une meilleure cohérence et traçabilité entre les modèles, ainsi qu'une plus grande facilité et rapidité de construction. Nous avons défini une méthode de traduction de modèles SysML en AltaRica Data Flow, permettant d'automatiser partiellement la création des modèles de SdF. Cette création est associée à l'utilisation de la BCD support de MéDISIS, permettant d'adjoindre la partie dysfonctionnelle du système, identifiée au cours de l'AMDEC.

La BCD utilisée au cœur de ces deux étapes nous permet de montrer comment modéliser les aspects d'un comportement dysfonctionnel en SysML. Cette étude est primordiale dans la volonté d'intégrer les notions de SdF au processus d'ISBM. Nous proposons des constructions permettant d'obtenir des bases de connaissances cohérentes avec les modèles des concepteurs, constituant un format de données pérennes à travers les différents projets de développement que peut mener un industriel. Cette BCD exploite le langage SysML, elle prouve la capacité du langage à modéliser toutes les dimensions d'un système, le rendant ainsi pleinement exploitable pour décrire des modèles orchestrant toutes les activités relatives à l'ISBM.

Les avantages du déploiement d'une telle méthode et de tels outils sont nombreux. La prise en compte des aspects SdF au cours du processus de conception, est améliorée en termes de traçabilité et de cohérence entre modèles et exigences. Le traitement des contraintes de SdF est rendu plus efficace car plus rapide, répétable, expressif et réutilisable. MéDISIS et les outils utilisés pour la déployer permettent la définition d'un environnement de développement système efficace permettant de traiter la SdF au cours d'un projet de conception dirigé par les modèles. Ces mécanismes permettent, à travers l'exploitation rigoureuse des modèles, de réaliser des études pérennes et de qualité.

Références

M. Cantor. Rational Unified Process® for Systems Engineering, RUP SE® Version 2.0. IBM Rational Software white paper, IBM Corporation, 8 mai 2003.

R. Chenouard, L. Granvilliers & R. Soto. Model-Driven Constraint Programming. Proceedings of 10th International Symposium on Principles and Practice of Declarative Programming, Valence (Espagne), 15-17 Juillet, 2008.

P. David, V. Idasiak & F.Kratz (2009a). Use and improvements of SysML in reliability study. Proceedings of the 55th Annual Reliability and Maintainability Symposium, RAMS 2009, Fort Worth, Texas, USA, Jan. 2009.

P. David, V. Idasiak & F.Kratz (2009b). Automating the synthesis of AltaRica Data-Flow models from SysML. Proceedings of ESREL 2009, Prague, République Tchèque, 7-10 septembre 2009.

P. David, Contribution à l'analyse de sûreté de fonctionnement des systèmes complexes en phase de conception : application à l'évaluation des missions d'un réseau de capteurs de présence humaine, Thèse de l'Université d'Orléans, soutenue le 9 novembre 2009 à l'ENSI de Bourges

P. David, V. Idasiak & F.Kratz, Reliability study of complex physical systems using SysML, Reliability Engineering and System Safety – Elsevier, Vol. 95 (april), pp. 431-450, 2010.

B. Douglass. The Harmony Process. I-Logix white paper, I-Logix, Inc., 25 mars 2005.

X. Dumas, C. Pagetti, L. Sagaspe, P. Bieber & P. Dhaussy. Vers le génération de modèles de sûreté de fonctionnement. 15ème Conférence francophone sur les Langages et Modèles à Objets (LMO) & Architectures Logicielles (CAL), Montréal, 5-7 mars 2008.

J. Estefan. Survey of Model-Based Systems Engineering (MBSE) Methodologies, Rev. B. INCOSE MBSE Initiative, 23 Mai 2008.

S. Friedenthal, R. Griego & M. Sampson. INCOSE MBSE Initiative. International Council on Systems Engineering 2007, San Diego, 24-29 juin 2007.

S. Friedenthal, A. Moore & R. Steiner. A Practical Guide to SysML : The Systems Modeling Language. The MK/OMG press, Elsevier, 2008.

H-P Hoffmann. Harmony-SE/SysML Deskbook: Model-Based Systems Engineering with Rhapsody. Telelogic/I-Logix white paper, Telelogic AB, 24 mai 2006.

International Council on Systems Engineering. Systems Engineering Vision 2020. Version 2.03, TP-2004-004-02, Septembre 2007.

F. Jouault & I. Kurtev. Transforming Models with ATL. Proceedings of 8th International Conference on Model Driven Engineering Languages and Systems (MODELS Satellite Events 2005), LNCS, Vol. 3844, pp. 128-138, 3 octobre 2005.

H. Lykins, S. Friedenthal, A. Meilich. Adapting UML for an Object-Oriented systems engineering method (OOSEM). In: Proceedings of the 10th annual INCOSE symposium, Juillet 2000.

P.-A. Muller, F. Fleurey & J.-M. Jézéquel. Weaving executability into object-oriented meta-languages. In Proceedings of MODELS/UML 2005, 2005

Object Management Group. OMG Systems Modeling Language (OMG SysML) V1.1. 1er novembre 2008.

A. Rauzy, Z. Brik & E. Arbaretier. Sûreté de Fonctionnement et Analyse de Performance. Actes du 16ème Congrès Lambda Mu : les nouveaux défis de la maîtrise des risques, Avignon, France, 7-9 octobre 2008.